The with statement:

- The with statement was introduced in python 2.5.
- The with statement is useful in the case of manipulating the files.
- It is used in the scenario where a pair of statements is to be executed with a block of code in between.

Syntax:

The syntax to open a file using "with", the statement is given below.

```
with   open(<file name>, <access mode>)  as  <file-pointer>:
       statement-1
        statement-2
        statement-n
```

- The advantage of using "with" statement is that it provides the guarantee to close the file regardless of how the nested block exits.
- It is always suggestible to use the "with" statement in the case of files because, if the break, return, or exception occurs in the nested block of code then it automatically closes the file, we don't need to write the close() function.
- It doesn't let the file to corrupt.

Consider the following example.

Example:

```
with open("file.txt",'r') as f:
      content = f.read();
      print(content)
```

```
# Python code to illustrate with statement
```

Example:

```
with open("file.txt") as file:
   data = file.read()
   print(data)
```

Writing data into the file:

To write some text to a file, we need to open the file using the open method with one of the following access modes.

w :

- If given file name is not existing then it will be create a new file name with given file name in specified path location.
- If given file name is already existing then It will overwrite the given file name.
- By default the file pointer is available at the beginning of the file position.

a :

- If given file name is not existing then it will be create a new file name with given file name in specified path location.
- If given file name is already existing then It will append new data at end of the given file data.
- It will not overriding the existing data.
- By default the file pointer is available at the end of the file data.

x :

- If given file name is not existing then it will be create a new file name with given file name in specified path location.
- If given file name is already existing then It will throws exception like **FileExistsError.**
- By default the file pointer is available at the end of the file data.

Writing the data in to file:

```
a = open('Employees.txt','w')
a.write('Python is easy language')
a.close()
```

Example:

```
# open the file.txt in append mode. Create a new file if no such file exists.
fileptr = open("file2.txt", "w")
# appending the content to the file
fileptr.write('''Python is the modern day language. It makes things so simple.
                 It is the fastest-growing programing language''')
# closing the open file object
fileptr.close()
```

Output: File2.txt

```
Python is the modern-day language. It makes things so simple. It is the fastest
growing programming language.
```

Appending new data to the existing file

```
a = open('Employees.txt', 'a')
a.write('\nPython is also simple language\nPython is powerful language too')
a.close()
```

Read file data through "for loop"

We can read the file data using for loop. Consider the following example.

Example:

```
#open the file.txt in read mode. causes an error if no such file exists.
fileptr = open("file2.txt","r");
#running a for loop
for  I  in  fileptr:
   print(i, end="") # i contains each line of the file
```

Output:

```
Python is the modern day language.
It makes things so simple.
Python has easy syntax and user-friendly interaction.
```

Note : We can also use write function along with **with** statement.

```
# Python code to illustrate with()  alongwith write()
with open("file.txt", "w") as f:
       f.write("Hello World!!!")
```

split() using files:

- We can also split lines using file handling in Python. This splits the file object data when space is encountered.
- You can also split using any characters as we wish. Here is the code.

Example:

```
fileObject = open('demo.txt', 'r')
data = fileObject.read()
list_of_words = data.split()
```

```
print("List of Words in a File :")
print(list_of_words)
print('Number of words in a file :')
print(len(list_of_words))
```

Output:

```
List of Words in a File :
['python', 'is', 'easy', 'language', 'python', 'is', 'more', 'powerfull', 'language',
'python', 'is', 'dynamic']
Number of words in a file :
12
```

splitlines() using Files

Example:

```
fileObject = open('demo.txt', 'r')
data = fileObject.read()
list_of_lines = data.splitlines()
print('List of lines in a file :')
print(list_of_lines)
```

Output:

```
List of lines in a file :
['python is easy language', 'python is more powerfull language', 'python is
dynamic']
Number of lines in a file :
3
```

Practice Examples:

Q. Write a program to displaying all the data from the file ?

```
A = open('Employees.txt','r')
x = a.read()
print(x)
```

or

```
print(open('Employees.txt','r').read())
```

Output:

```
tcs,narayana,venu,balu
wipro,suresh,naresh,Ramesh
info,renu,amala,kavya,ramu
```

```
cts,veni,satya,durga,ramya
```

Q. Write a program to displaying first line from the file ?

```
A = open('Employees.txt','r')
v=a.readline()
print(v)
or
print(open('Employees.txt','r').readline())
or
a=open('Employees.txt','r')
v=a.readlines()
print(v[0])
or
print(open('Employees.txt','r').readlines()[0])
```

Output: tcs, narayana, venu, balu

Q. Write a program to displaying second line from the file?

```
a = open('Employees.txt','r')
v = a.readlines()
print(v[1])
or
print(open('Employees.txt','r').readlines()[1])
```

Output:

```
wipro,suresh,naresh,Ramesh
```

Q. Write a program to displaying 3rd line from the file ?

```
a=open('Employees.txt')
v=a.readlines()
print(v[2])
or
print(open('Employees.txt').readlines()[2])
```

Output:

```
info,renu,amala,kavya,ramu
```

Q. Write a program to displaying last line in the file?

```
a = open('Employees.txt','r')
v = a.readlines()
print(v[-1])
```

or

```
print(open('Employees.txt','r').readlines()[-1])
```

Output: cts,veni,satya,durga,ramya

Q. Write a program to find number of lines in the file ?

```
a = open('Employees.txt','r')
v = a.readlines()
print(len(v))
```

Output: 4

Working with Words:

Q. Write a program to displaying first word from each line?

```
a = open('Employees.txt','r')
v = a.readlines()
for  i  in  v:
    k = i.split(',')
    print(k[0])
```

or

```
for i in open('Employees.txt','r').readlines():
    print(i.split(',')[0])
```

Output:

```
tcs
wipro
info
cts
```

Q. Write a program to displaying second word from each line?

```
a = open('Employees.txt','r')

v = a.readlines()
```

```
for  i  in  v:
     k = i.split(',')
     print(k[1])
```

or

```
for i in open('Employees.txt','r').readlines():
     print(i.split(',')[1])
```

Output:

```
Narayana
suresh
renu
veni
```

Q. Write a program to displaying last word from the each line?

```
a = open('Employees.txt','r')
v = a.readlines()
for i in v:
     k = i.split(',')
     print(k[-1])
```

or

```
for  i  in open('Employees.txt','r').readlines():
     print(i.split(',')[-1])
```

Output:

```
balu
Ramesh
ramu
ramya
```

Q. Write a program to displaying number of characters in the file(including commas)?

```
a = open('Employees.txt','r')
v = a.read()
print(len(v))
```

Output : 103

Q. Write a program to displaying number of characters in the file(excluding commas)?

```
a = open('Employees.txt','r')
v = a.readlines()
c = 0
for  i  in v:
   x = i.split(',')
   for j in x:
      c = c + len(j)
print(c)
```

Output: 89

Q. Write a program to Counting number of vowels in the file ?

```
obj = open('sample.txt','r')
#print(obj)
data = obj.read()
vowels = 'aeiouAEIOU'
d = {}.fromkeys(vowels,0)
#print(d)
for char in data:
   if char in d:
      d[char]= d[char]+1
print(d)
```

Output: {'a': 15, 'e': 6, 'i': 5, 'o': 2, 'u': 6, 'A': 0, 'E': 0, 'I': 0, 'O': 0, 'U': 0}

Q. How to find each word count how many times coming from given file ?

```
obj = open('demo.txt','r')
data = obj.read()
words = data.split()
d = {}
for word  in words:
   if word in d:
      d[word]= d[word] + 1
   else:
      d[word]= 1
```

```
print(d)
```

Output: {'Pthon': 1, 'is': 3, 'a': 1, 'language': 1, 'python': 2, 'easy': 1, 'powerful': 1}

Q. Write a program to display filepointer location and modify file pointer locations.?

Example:

```
x = open("test.txt")
print(x.tell())   ----->> 0
data = x.read(5)
print(data)    --------------> srini
print(x.tell())-----------> 5
data1 = x.readline()
print(data1)---------->vas,j
print(x.tell())------->  13
x.seek(0)
print(x.tell())-------->0
data2 =x.readline()
print(data2)---------->srinivas j
print(x.tell())------->13
```

Q) Write the code into file using Exception handling :

```
fileObject = None
try:
    fileObject = open('hello4.txt', 'w')
    fileObject.writelines('Python is  easy language')
    print('data is stored in a file')
except:
    print('error occured while writing data into a file')
finally:
    if fileObject:
        fileObject.close()
        print('file is closed')
print(fileObject.read()) # ValueError: I/O operation on closed file.
```

Output:

```
data is stored in a file
file is closed
Traceback (most recent call last):
  File "D:/Python@VLR/myfile.py", line 196, in <module>
    print(fileObject.read()) # ValueError: I/O operation on closed file.
ValueError: I/O operation on closed file.
```

============= *********   ===========   **************